NAVAL POSTGRADUATE SCHOOL
Monterey, California

THESIS

OPTICAL SYSTEM EVALUATION

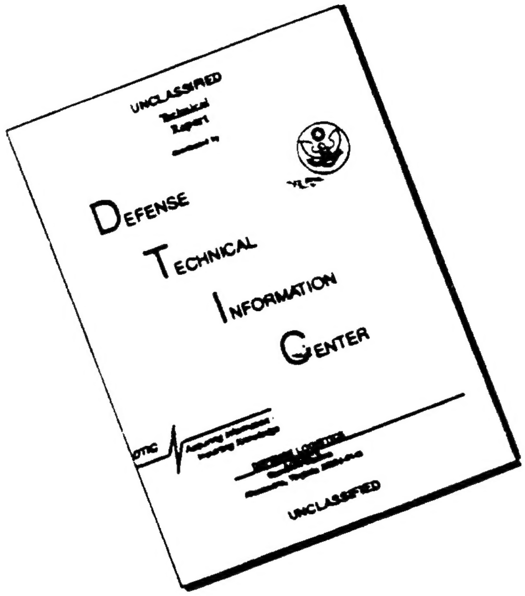
Carlos Renato Campos Rangel

December 1995

Thesis Advisors: David Scott Davis
Gordon Everett Schacher

Approved for public release; distribution is unlimited.

DISCLAIMER NOTICE



THIS DOCUMENT IS BEST QUALITY AVAILABLE. THE COPY FURNISHED TO DTIC CONTAINED A SIGNIFICANT NUMBER OF PAGES WHICH DO NOT REPRODUCE LEGIBLY.

REPORT DOCUMENTATION PAGE

Form Approved
OMB No. 0704-0188

1. AGENCY USE (Leave Blank)	2. REPORT DATE	3. REPORT TYPE AND DATES COVERED
	Dec 1995	Master's Thesis

4. TITLE AND SUBTITLE

OPTICAL SYSTEM EVALUATION

5. FUNDING NUMBERS

6. AUTHOR(S)

Rangel, Carlos Renato Campos

7. PERFORMING ORGANIZATION NAME(S) AND ADRESS(ES)

Naval Postgraduate School
Monterey, CA 93943-5000

8. PERFORMING ORGANIZATION REPORT NUMBER

9. SPONSORING / MONITORING AGENCY NAME(S) AND ADDRESS(ES)

10. SPONSORING / MONITORING AGENCY REPORT NUMBER

11. SUPPLEMENTARY NOTES

The views expressed in this thesis are those of the author and do not reflect the official policy or position of the Department of Defense or the United States Government.

12a. DISTRIBUTION / AVAILABILITY STATEMENT

Approved for public release; distribution is unlimited.

12b. DISTRIBUTION CODE

13. ABSTRACT *(Maximum 200 words)*

Optical and infrared sensors have an important role to play in modern military engagements, as the deployment of passive systems increases. To guarantee the efficient development and usage of such equipment, at a reasonable cost, a reliable and realistic simulation of sensor performance is fundamental. The research project presented in this thesis consists of two parts. First, basic software modules that characterize the target-detector radiative transfer problem were developed. This was accomplished by developing separate modules for each physical aspect of the problem. The second part concerned the viability of implementing the physics of such real-world radiative transfer effects into existing military simulation tools. The chosen simulation environment for this effort was *NPS Platform Foundation*, an existing simulating software package that was developed at the Naval Postgraduate School.

14. SUBJECT TERMS	15. NUMBER OF PAGES
optical sensors, target-detector radiation, simulation	52
	16. PRICE CODE

17. SECURITY CLASSIFICATION OF REPORT	18. SECURITY CLASSIFICATION OF THIS PAGE	19. SECURITY CLASSIFICATION OF ABSTRACT	20. LIMITATION OF ABSTRACT
Unclassified	Unclassified	Unclassified	UL

NSN 7540-01-280-5500

Standard Form 298 (Rev. 2-89)
Prescribed by ANSI Std. 239-18

OPTICAL SYSTEM EVALUATION

Carlos Renato Campos Rangel
LCDR, Brazilian Navy
B.S., Brazilian Naval Academy, 1983

Submitted in partial fulfillment of the
requirements for the degree of

MASTER OF SCIENCE IN APPLIED PHYSICS

from the

NAVAL POSTGRADUATE SCHOOL
December 1995

Author: ______________________________
Carlos Renato Campos Rangel

Approved by: ______________________________
David Scott Davis, Thesis Advisor

____
Gordon Everett Schacher, Thesis Advisor

____
William B. Colson, Chairman
Department of Physics

ABSTRACT

Optical and infrared sensors have an important role to play in modern military engagements, as the deployment of passive systems increases. To guarantee the efficient development and usage of such equipment, at a reasonable cost, a reliable and realistic simulation of sensor performance is fundamental. The research project presented in this thesis consists of two parts. First, basic software modules that characterize the target-detector radiative transfer problem were developed. This was accomplished by developing separate modules for each physical aspect of the problem. The second part concerned the viability of implementing the physics of such real-world radiative transfer effects into existing military simulation tools. The chosen simulation environment for this thesis was *NPS Platform Foundation*, an existing simulation software package that was developed at the Naval Postgraduate School.

TABLE OF CONTENTS

I. INTRODUCTION

Optical radiation sensors and their associated detector systems now play an indispensable role in modern military systems. This is particularly true of sensor systems that operate at visible and infrared wavelengths. Typical uses of such systems include target detection, target tracking, target identification-friend-or-foe(IFF), passive surveillance, scene background monitoring, and so forth.

Such modern systems consist of at least two basic subsystems, or components. The first of these is the optical sensor itself, which receives incoming electromagnetic radiation and converts, or transduces, that flux of radiation into an electrical signal. The second subsystem is comprised of one or more devices that process the electrical signal into whatever form is appropriate for the given military application. For many types of systems, such as trackers and seekers, there are additional subsystems that take the processed electrical signal and use it to drive some other device, such as a suite of servomotors that control the pointing of the tracker or the flight attitude of the seeker.

Most modern systems employ some sort of solid state detector(s) as their front-end transducers, and sophisticated signal processing electronics, based upon built-in computers, as the processing stage.

The purpose of this thesis research is to develop a set of computer program modules that accurately model, to first order, the behavior of an infrared target detection system under realistic conditions. Because of the great time and expense involved in actually building prototype military systems, initial

simulations can pinpoint potential pitfalls in a prototype long before it goes into the hardware production phase of development.

This project has been addressed in two phases. The first phase develops basic software modules for target and detector characterization in their pertinent environment, based upon the physics of the radiative transfer problem. These modules were written in ANSI standard C. The second phase of the project investigates how basic physics issues (including the optical radiation) can be implemented in an object-oriented simulation of a target-detector system in the future.

The development of this project has proceeded in the following manner, as outlined below and discussed in ensuing chapters, as indicated:

- Chapter II - The Physics of Radiation Detection;
- Chapter III - Graphical Examples of Radiative Transfer Calculations;
- Chapter IV - Discussion of Basic Physics Implementation in *NPS Platform Foundation*;
- Chapter V - Conclusions;
- Appendix A - The C Program Modules;
- Appendix B - The *NPS Platform Foundation* Implementations.

II. THE PHYSICS OF RADIATION DETECTION

In order to simulate the performance of optical detection systems and their incorporation into military systems, it is necessary to understand the basic physics of the production, transport and detection of optical radiation. The branch of optical physics that deals with these matters is called *radiometry* and/or *radiative transfer* (Dereniak, 1984; Boyd, 1983; Chandrasekhar, 1960; Born and Wolf, 1980) when the radiation field is involved, while *detector physics* is the discipline that describes detection devices themselves. This chapter summarizes the aspects of those disciplines that are relevant to the research involved in this thesis project.

A. THE PRODUCTION OF OPTICAL RADIATION

In general, the term *optical radiation* is a broad one, used to label electromagnetic radiation whose wavelength lies approximately in the 100 nm - 100 μm range. Most detector systems employed in military hardware operate at either visible (400 - 700 nm) or near-to-mid-infrared (700 nm - 12 μm) wavelengths. There are several natural and man-made mechanisms that generate such radiation. For this thesis project, we are interested in thermal radiation in the infrared portion of the electromagnetic spectrum.

All objects in the universe exist at temperatures above absolute zero. By virtue of their non-zero temperatures, all such objects emit electromagnetic radiation. As the object's temperatures increases, the total amount of emitted

radiation increases, and the frequency at which the peak of the radiation distribution occurs shifts to higher values, corresponding to shorter wavelengths. The basic process is that the atoms and molecules that compose the objects move faster at higher temperatures, constrained by interactions with other atoms and molecules. Therefore, the elementary charges within these atoms are subjected to accelerations which generate electromagnetic radiation.

From basic thermodynamic theory it can be shown that an ideal, perfect emitter of radiation must also be a perfect absorber. Since a perfect absorber, at low temperature, would appear black, it is therefore called a *blackbody*. Blackbody sources can be good approximations to actual infrared emitters of military interest.

The geometry of the radiation emission process is illustrated in Figure 2.1

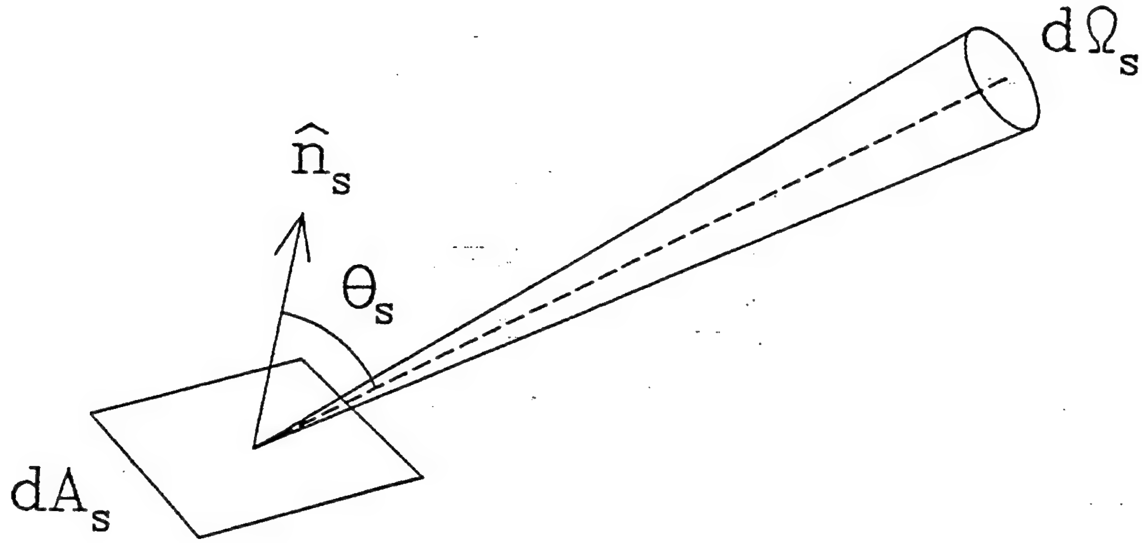


Figure 2.1. The Radiation Emission Process.

A small area element dA_s (s denotes a radiation source) emits radiation into the hemispherical region beyond the surface. This hemisphere can be partitioned into a set of infinitesimal solid angles $d\Omega_s$, each of which corresponds to a small channel for the outgoing radiation to follow as it propagates. The actual amount of radiation that flows into $d\Omega_s$ is proportional to both $d\Omega_s$ and the projected area of the radiating surface in the propagation direction, or $(dA_s)(\cos\theta_s)$. θ_s is measured between the surface normal vector $\hat{n}_s$ and the propagation direction.

The radiation field then propagates through the intervening atmosphere toward the detector. Although the atmosphere will actually affect the radiation in very complicated ways, such as absorption, scattering, etc. (Goody and Yung, 1989), such effects will be neglected for purposes of this thesis research. Such a simplification is reasonably valid in clear atmospheres whenever the wavelengths of interest are far removed from atmospheric atomic and molecular absorption features.

The detector optics usually consists of an input aperture and lens, comprising a small telescope (Hecht, 1989) which collects the incident radiation and focuses it onto the detector. The geometry of the system is shown in Figure 2.2 .

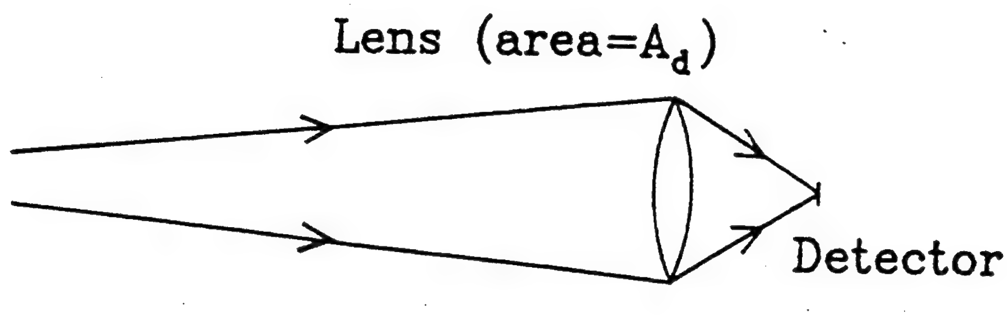


Figure 2.2. Detector Optics.

The collecting lens can be partitioned into small area elements of size dA_d as shown in Figure 2.3 .

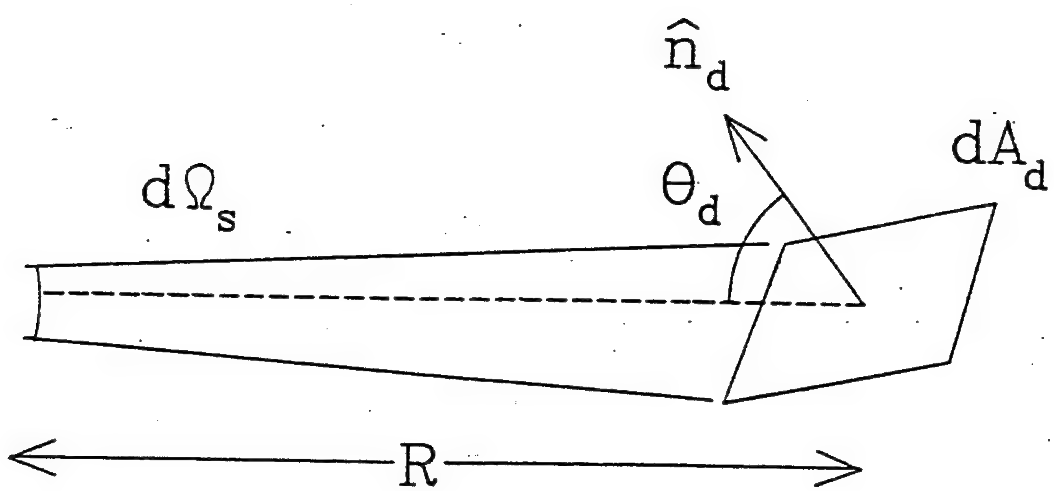


Figure 2.3. Collecting Lens' Subpartition.

Notice from the geometry of the layout that the solid angle subtended by the detector area element dA_d, as seen by the source, is

$$d\Omega_s = \frac{(dA_d)\,(\cos\theta_d)}{R^2}, \qquad (2.1)$$

where R is the source-detector range and θ_d is measured between the surface normal vector $\hat{n}_d$ and the source-detector line of sight. When this is combined with the results of the previous discussion, it follows that the detected signal, called the flux $d\Phi$, is proportional to $\{(dA_s)(\cos\theta_s)(dA_d)(\cos\theta_d)\} \div R^2$, or

$$d\Phi = \frac{L\cos\theta_s\cos\theta_d\,dA_s\,dA_d}{R^2} \qquad (2.2)$$

The quantity L is called the radiance in units of watt per steradian per square meter. If the source is sufficiently well collimated that all of the emitted radiation falls on the detector, then only the power emitted by the source needs to be known. However, for our purpose, the detector only intercepts a small fraction of the radiated signal (Dereniak, 1984).

Our source is considered to be approximately Lambertian (Dereniak, 1984; Boyd, 1983; Born and Wolf, 1980). A Lambertian surface is one for which the surface radiance is independent of the angle from which is viewed. Alternately, a Lambertian radiator is an isotropically diffuse surface for which the

radiant intensity in any direction varies as the cosine of the angle (θ_s) between that direction (detector-target line) and the normal to the surface (target). Then, if we multiply L by π (which would be the result of integrating L over one hemisphere for the Lambertian source), we get M_α, which is called the exitance (Dereniak, 1984). The subscript α is explained below.

Optical sensors systems employ detectors that can be classified into two broad categories. One type is sensitive only to the total incident radiant power, or rate of energy transferred to the detector. Those are designated as power or power-sensing detectors. The other category of detector is called a quantum or photon detector. Such detectors sense radiation quanta, or photons. Because of these two detector types, the subscript α is used to denote that M_α is either the photon flux exitance ($\alpha \rightarrow p$) used with quantum detectors, or the radiant exitance ($\alpha \rightarrow e$) as appropriate for power detectors. In what follows in this thesis, M_α will be used to represent both types of exitances, M_p (in photons per second per square meter) for photon flux exitance, and M_e (in watts per square meter) for radiant exitance. Using that convention, Equation 2.2 is written in terms of M_α as

$$d\Phi = \frac{M_\alpha}{\pi} \frac{\cos\theta_s \cos\theta_d \, dA_s \, dA_d}{R^2}. \qquad (2.3)$$

For purposes of this thesis research, we assumed that the detector's surface normal always coincides with the source-detector line. In other words, θ_d is equal to zero. This is generally the case for the narrow field of view detector optics that are normally employed in remote sensing, target acquisition, and tracking applications. With that simplification, Equation 2.3 becomes:

$$d\Phi = \frac{M_\alpha}{\pi} \frac{\cos\theta_s \, dA_s \, dA_d}{R^2}. \tag{2.4}$$

B. PLANCK'S DISTRIBUTION FOR BLACKBODY RADIATION

The electromagnetic power radiated by a blackbody is given by the Planck radiation distribution (Dereniak, 1984):

$$M_{e,\lambda}(\lambda, T) = \frac{2\pi hc^2}{\lambda^5 \left[e^{\frac{hc}{\lambda kT}} - 1 \right]} \tag{2.5}$$

where $M_{e,\lambda}$ is the spectral radiant exitance in watts per square meter of radiation surface area per wavelength interval (m).

T is the absolute temperature of the blackbody in (K),

h is Planck's constant (6.626176×10^{-34} J s),

λ is the emitted wavelength in meters (m),

c is the speed of light (2.99792438×10^{8} m s^{-1}), and

k is Boltzman's constant (1.380662×10^{-23} J K^{-1}).

The total emission in any spectral region of interest can be obtained by integrating Equation (2.5) over the wavelength bandwidth of interest:

$$M_e = \int_{\lambda_1}^{\lambda_2} \frac{2\pi hc^2}{\lambda^5 \left[e^{\frac{hc}{\lambda kT}} - 1 \right]} d\lambda, \qquad (2.6)$$

where λ_1 and λ_2 are the lower and upper wavelength limits of integration, respectively, and the resulting M_e is the radiant exitance, in Wm^{-2} . It is evident that this integral is of paramount importance when simulating or modeling the behavior of infrared power-detecting sensor systems, because $M_{e,\lambda}$ peaks at about λ=10μm for typical ambient temperatures T$\approx$300K.

Since photon (quantum) detectors are also important, an expression for the corresponding spectral photon flux exitance is also needed. One can find such an expression by simple reasoning. Since a photon of wavelength λ transports energy $Q=hc/\lambda$, and $M_{e,\lambda}$ is the radiated power per area per wavelength interval, it follows that $M_{p,\lambda} = M_{e,\lambda}/(hc/\lambda)$. Hence:

$$M_{p,\lambda}(\lambda, T) = \frac{2\pi c}{\lambda^4 \left[e^{\frac{hc}{\lambda kT}} - 1 \right]}, \qquad (2.7)$$

where $M_{p,\lambda}(\lambda,T)$ is the photon flux exitance at a particular wavelength λ, in photons per second per square meter per wavelength interval, in meters.

The total photon exitance at a particular bandwidth is found by integrating Equation 2.7 over the region of interest:

$$M_p = \int_{\lambda_1}^{\lambda_2} \frac{2\pi c}{\lambda^4\left[e^{\frac{hc}{\lambda kT}} - 1\right]} d\lambda. \tag{2.8}$$

C. EMISSIVITY

Emissivity (Dereniak, 1984) is introduced to quantify the different emission properties of different objects. This is needed, because no actual source radiates like a perfect blackbody. The spectral emissivity is defined to be the ratio of the radiance of the object under consideration to that of a perfect blackbody:

$$\varepsilon_e(\theta,\phi,\lambda) = \frac{L_{e,\lambda}(\lambda,T)_{actual}}{L_{e,\lambda}(\lambda,T)_{blackbody}}, \tag{2.9}$$

where $L_{e,\lambda}$ is the radiance in watts per steradian per square meter, and θ and ϕ define the direction of the radiated beam with respect to the surface normal vector of the radiator.

Note that $0 \leq \varepsilon_e(\theta,\phi,\lambda) \leq 1$. The average emissivity over the entire spectrum is then

$$\varepsilon_e = \frac{\int_0^\infty \varepsilon_e(\lambda) L_{e,\lambda}(\lambda,T) d\lambda}{\int_0^\infty L_{e,\lambda}(\lambda,T) d\lambda}. \tag{2.10}$$

The temperature dependence of ε_e is omitted only for notational simplicity, although the definition clearly implies that ε_e is temperature dependent.

The photon average emissivity, on the other hand, is

$$\varepsilon_p = \frac{\int_0^\infty \varepsilon_p(\lambda)\ L_{p,\lambda}(\lambda,T) d\lambda}{\int_0^\infty L_{p,\lambda}(\lambda,T) d\lambda}, \tag{2.11}$$

where $L_{p,\lambda}$ is the photon flux radiance (sterance) in photons per second per steradian per square meter.

For practical applications, whenever the photon emissivity of an object is needed, it can be obtained from tabulations of relevant data (Wolfe and Zissis, 1989).

D. RADIATION NOISE

The detected signal from a quantum detector is simply the number of photons that it detects in a given time interval. However, there is uncertainty, or fluctuation, in the amount of electromagnetic radiation emitted by any signal source within a given time interval. This uncertainty is called photon, or source, radiation noise (Dereniak, 1984). Since thermal radiators obey Poisson statistics, the root mean square (rms) noise level is given by the square root of the number of photons emitted in a period of time:

$$\text{Noise}_{rms} = \sqrt{M_p . t} \qquad (2.12)$$

where M_p is the total photon flux exitance over the bandwidth in photons per second per square meter, and t is the measurement or observation time in seconds.

E. POWER NOISE

The dominant noise source in a power detector is called shot noise, which is simply the detector's fluctuating output in response to the photon noise that it receives. Shot noise is a function of the detector's *responsivity*, which is a basic figure of merit that applies to all detectors with electrical output. Responsivity ($\Re_i$) is the ratio of the output (in our case, current in amperes) to the radiant input (power flux, in watts) :

$$\Re_i = \frac{i_{signal}}{\Phi_e}, \tag{2.13}$$

where i_{signal} is the current signal output, in amperes, and Φ_e is the power flux, in watts.

Shot noise is defined to be the root mean square (rms) noise current (i_{rms}) given by:

$$i_{rms} = i_{noise} = \sqrt{\frac{q\,\Re_i \Phi_e}{t}}, \tag{2.14}$$

where q is the elementary charge constant (1.60×10^{-19}C),

$\Re_i$ is the responsivity, in amperes per watt,

Φ_e is the power flux, in watts, and

t is the observation time, in seconds.

The basic quantity of interest in characterizing a detector system's performance is its signal-to-noise ratio (SNR). Given the optical geometry and detector characteristics of a particular target-detector system configuration, the system's SNR can be predicted using Equations 2.4, 2.12, and 2.14. These considerations form the foundation for the development of the C program modules that were written to characterize the infrared detection problem.

III. GRAPHICAL EXAMPLES OF RADIATIVE TRANSFER CALCULATIONS

A set of easily portable software routines was written to implement the calculations outlined in Chapter II. The intent was to develop a package of routines that could be easily incorporated into an object-oriented simulation system. The routines were written in ANSI standard C.

Appendix A lists the C program functions that were produced. Examples of their utility are illustrated in Figures 3.1 and 3.2. In the following paragraphs, the acronyms between parentheses denote how the variables were named in the programs.

Figure 3.1 shows the flux output for a photon detector, with working wavelength bandwidth of 3.0 (la) to 5.0 (lb) $\times 10^{-6}$ meters. It represents the photon flux expected from a $1m^2$ target area (Sa), located 100 meters (range) away from a sensor system with a $1m^2$ collecting area (Da), as a function of the angle between the target's plane normal and the source (target)-detector line (Steta), in degrees. The angle (Steta) varies from zero to ninety degrees. The target does not overfill the field of view, and its temperature (vinkel) is 300 K.

Figure 3.2 shows the resultant photon flux for the same working bandwidth detector, as a function of the distance from the telescope (range), which varies from 10 to 100 meters. All others parameters are kept the same as in Figure 3.1, and the target orientation (Steta) is assumed to be normal to the line of sight, so that Steta is equal to zero.

These figures are only illustrative of the basic capabilities of the software modules. The modules themselves can be incorporated directly into more complex and realistic simulation systems.

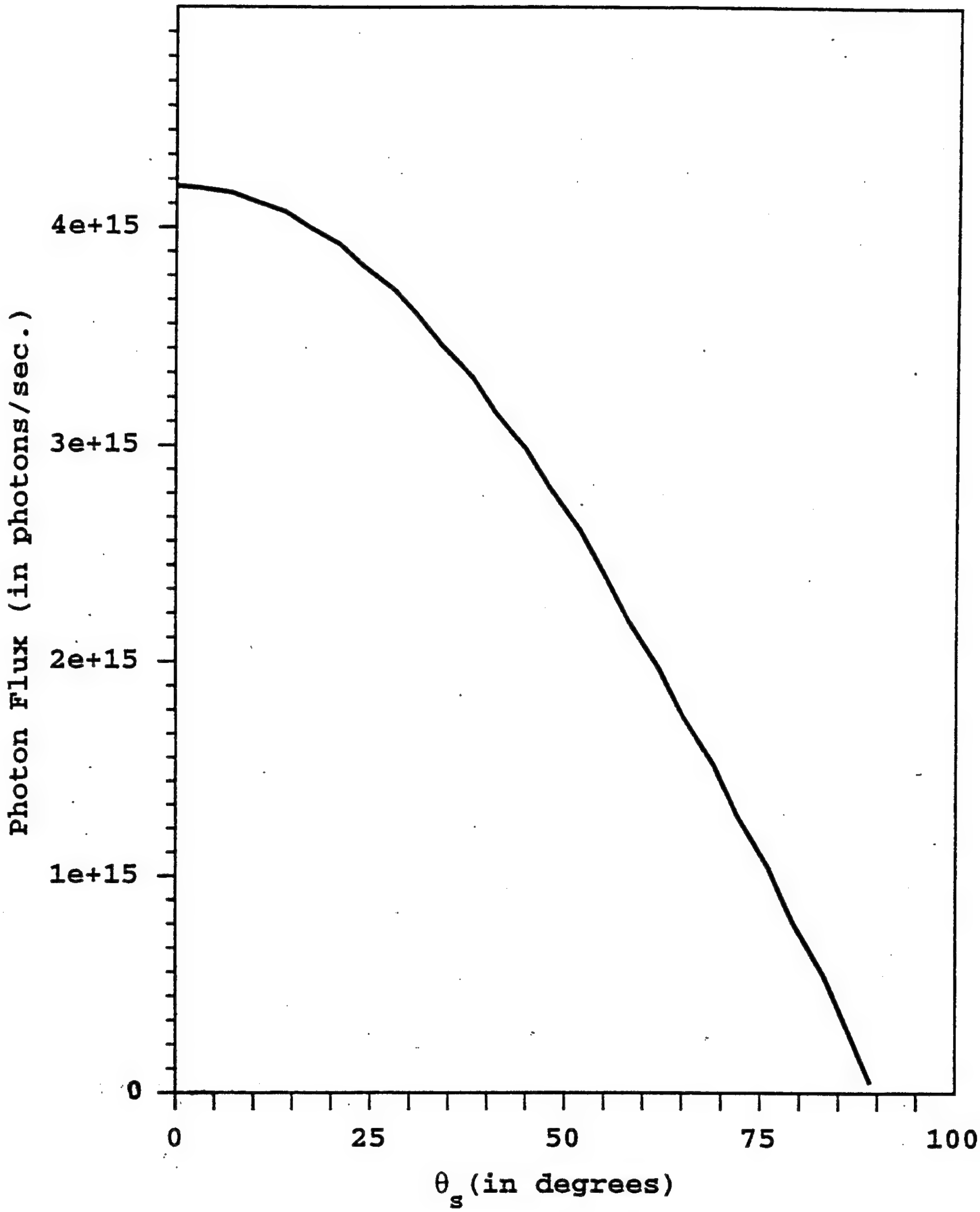


Figure 3.1. Graph of Photon Flux as a Function of θ_s.

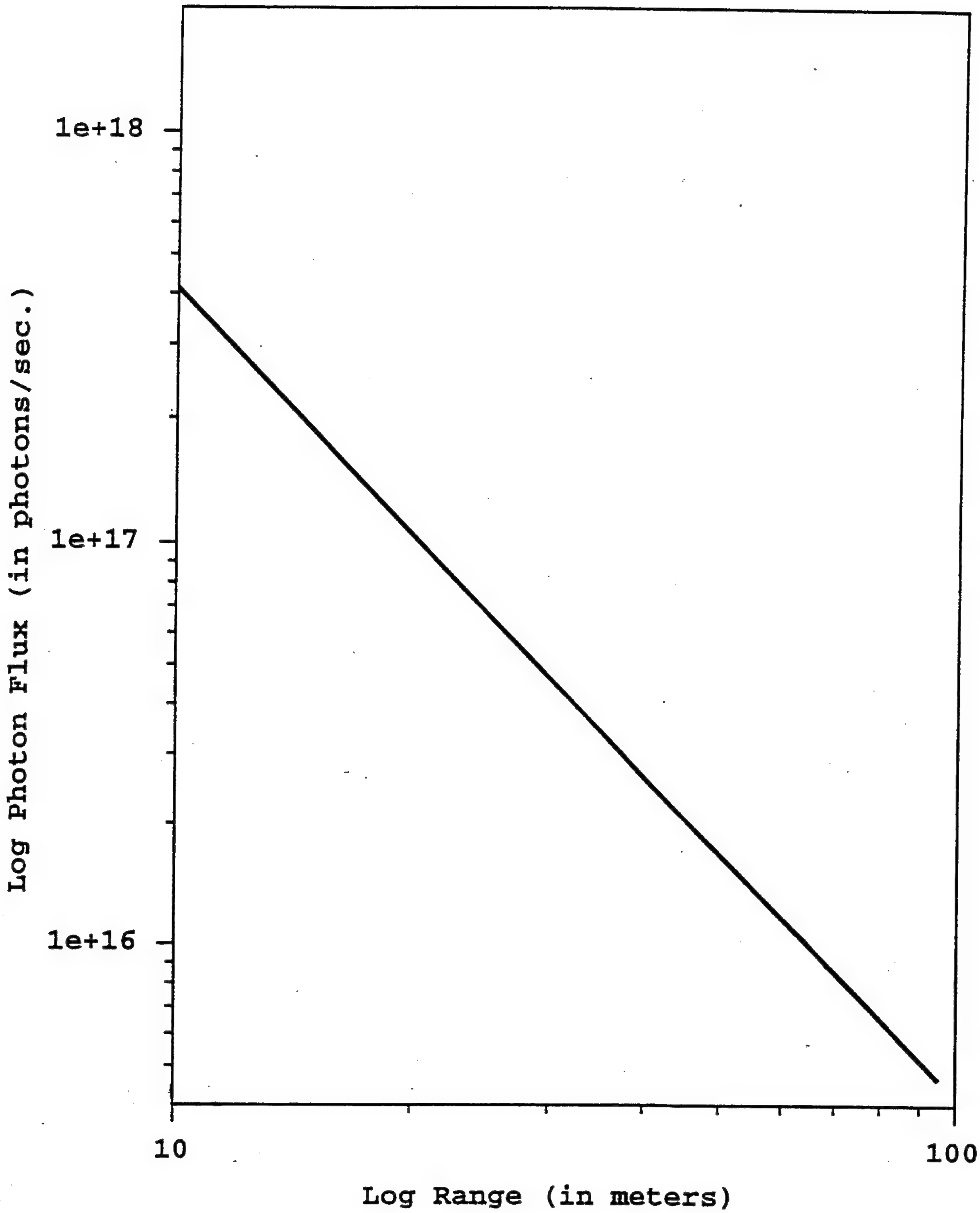


Figure 3.2. Graph of Photon Flux as a Function of R.

IV. DISCUSSION OF BASIC PHYSICS IMPLEMENTATION IN *NPS PLATFORM FOUNDATION*

Computer simulation is a fundamental method for studying complex systems. It consists of designing a model of an actual or theoretical physical system, and executing the model on a digital computer in such a manner that it replicates accurately the behavior of the system. The purpose can vary from a simple observation of system behavior to training personnel, to a complex analysis of all parts of the system and human in-the-loop interactions (Fishwick, 1994). Unique aspects of military simulations are to assist in designing tactics and to train personnel in real-time decision making in a complex environment.

There are many well-adapted commercial simulation tools for specific types of problems, for example, communications programs which have pre-programmed transmitters, receivers, processors, links, and protocols; assembly line programs which have available workstations, partially assembled pieces, and conveyers. These simulation software packages allow analysts to generate quickly and easily sophisticated simulations that replicate their specific processes (Bailey, 1994).

Military operations analysts work in a domain where the key components are platforms, weapons, sensors and tactics. At the time of this writing, there is no commercial, off-the-shelf simulation software product which is designed to model the military environment. This is understandable, since military systems are composed of an extremely wide range of system types, must operate in

varied environments, and are subject to many human interactions. The result is that military simulations have been expensive to build, and the focus of the military modeling and simulation community has been directed more toward software development than analysis (Bailey, 1994). Model development has focused mainly on hardware level systems. Simulation development has focused on large scale engagements (e.g., *MARS, EADSIM,* and *JANUS*), and less on smaller scale details of sub-system interactions (e.g., optical systems or radar performance).

Prior to building a military computer simulation, one question must be answered: "What is the purpose of the simulation?" If the purpose is to represent a theater level engagement over a large geographical area, a high fidelity representation for every object in the engagement becomes very lengthy for the computer to run. Computation time could turn out to be prohibitive. On the other hand, if the purpose is a one-on-one engagement, the physics fidelity of the two elements involved has a higher priority. Since only two objects are involved, the increased computation time to represent the physics correctly does not result in excessive run time. There will always be a trade-off between fidelity and computation time. The purpose of this portion of the thesis research is to represent the interaction between a detector and a target in a one-on-one engagement. Thus our interest is to include the physics as well as possible.

It must be emphasized that the goal of this section of the thesis research is not to build new simulation software of an optical sensor. Rather, we want to determine how to modify existing simulation software so that its target

detection/acquisition portion includes realistic characterizations of real-world environmental conditions and the influence of those conditions on sensors. We have available *MARS* and *NPS Platform Foundation* simulations, thus they are natural choices for this study. They are written in the simulation language MODSIM. It is a modular, object-oriented, general purpose, procedural programming language which has built-in graphics capability to perform discrete-event simulation. The language includes a standard support environment consisting of an automated compilation manager and a graphics editor which is used to design icons, dialog boxes, menus and other graphics objects. MODSIM was designed as a production quality language to be used by working programmers constructing large programs. It was originally intended to replace other languages being used in large-scale United States government software projects, but it has now come into much wider use. MODSIM allows C and C++ language code to be included as implementation modules where low-level programming capability is required. The language provides a clearly defined way to specify non-MODSIM routines in its definition modules. These C and C++ modules are automatically compiled and linked by the compilation manager when MODSIM's module and procedure naming conventions are followed (Belanger, 1993). One important characteristic of this object-oriented language is the inclusion of a comprehensive Input/Output (I/O) capability, allowing stream and random access to I/O files. This means that one can easily attach attributes to objects by simply changing/creating the I/O files for each specific object.

Although *MARS* is user-friendly, for our purpose it has the disadvantage that one cannot easily modify the characteristics of a platform or a sensor because it was built for large scale air defense scenarios. In other words, once the platform (e.g.,ship or aircraft) is chosen, its pre-assigned sensors are not available for modification. *NPS Platform Foundation* has been designed to allow easy alteration of the characteristics of a platform, thus is more advantageous for our work. Further, we can also export the *NPS Platform Foundation* results to *MARS.*

Platform Foundation is a tool for modeling military engagements, consisting of a collection of objects which provide the functionalities required to model situations where platforms interact using sensors, weapons, and tactics. It was written in over 17,000 lines of MODSIM, and 3,000 lines of C, and designed as part of a course in simulation at the Naval Postgraduate School. *Platform Foundation* objects (PlatformObjs) possess a set of generic capabilities which provide all that is needed to build simple simulation models. Simply by changing a set of data files, an analyst can mount weapons and sensors on platforms, and make the platforms execute complex maneuver sequences. All the action is displayed on a geographic situation display. Finally, each platform issues periodic situation updates using the Distributed Interactive Simulation protocol. All of these capabilities work in a more or less automatic way (Bailey, 1994).

The formal division between generic Foundation capabilities and functions which are programmed for a specific application, is called the *line*. In terms of software design, the objects below the line are the PlatformObj, the SensorObj, and the WeaponObj, and all of the scenario management and data collection software. PlatformObjs are things which move through space and time, using sensors and weapons to interact with other platforms. *NPS Platform Foundation* allows an analyst to concentrate on tactics and doctrine because it provides key capabilities which every platform needs. The intent is that objects below the line are incorporated in special-purpose, application-specific objects like radars, ships, aircraft, and missiles. These above-the-line objects can use below-the-line capabilities as primitive elements in their tactical methodology (Bailey, 1994). The main advantage of *Platform Foundation* over *MARS* is the flexibility assured by the "above/below-the-line" feature.

Also important is that *Platform Foundation* has available analysis capabilities. It can assist on system performance and easily do statistical analysis. This is done using two applications called STATWORKBENCH (integrated graphical statistical analysis tool) and SMMAT (simulated material mobility analysis toolbox) (Bailey, 1994).

Each platform is capable of housing an inventory of weapons. WeaponObjs are special platforms-they have sensors, they move, they are animated, have fuel capacities, and can themselves be detected, damaged or destroyed. Weapons have specially built maneuvers based not on geography, but on the position and movement of the designated target. Special animation

features of the generic Foundation platform allow animation of damage and destruction caused by weapon impacts. Weapon effects, while above-the-line, are implemented through the provided platform damage models (Bailey, 1994).

Each SensorObj possesses a collection of objects called virtual sensors. A VirtualSensorObj is created for each target platform that the sensor might possibly detect. The virtual sensor computes the time that the target platform it is associated with will enter and exit the sensor detection range and time of closest approach. These calculations are performed as a first step in the simulation, before the animation begins. These events are rescheduled each time that the target platform or the platform on which the sensor is mounted changes its path. These pre-calculations, although challenging to develop for each possible movement situation, are much more efficient than repeatedly checking each platform pair during simulation execution to determine if there should be a detection. One of the advantages of the object-oriented sensor architecture is that a different maximum detection range is determined for each sensor-platform pair. Each sensor, being a collection of virtual sensors, maintains a list of platforms along with the true movement status of each. Any error model which the modeler chooses to employ can be implemented in an above-the-line virtual sensor. Sensors are graphically represented as range rings. The ring is animated by changing color when a detection event occurs. Unfortunately, each virtual sensor in a sensor may have a different range, but the graphical representation of the sensor allows only a single range (Bailey, 1994).

Our purpose here is to modify the sensor calculation from a simple determination of a “cookie-cutter” range. The modification is due primarily to environmental effects. We will address optical line-of-sight sensors. The environmental effects include:

- optical horizon,
- background radiance,
- terrain,
- clouds, rain, fog, and smoke, and
- scintillation

We have chosen for this thesis to implement the horizon effect because it is the simplest, but allows us to set up a procedure for modifying the sensor calculation methodology. There are two ways to address this problem. Either the platform sensor range ring or the threat (target) range can be modified. For the first method, the ring radius would appear smaller (or greater), and for the target range method, the sensor's ring would change color when the target appeared above the horizon, not when the target passed the maximum range distance.

The methodology employed for the horizon effect can be partially applied to the issues of terrain and cloud (fog, smoke, and rain) effects. See Figure 4.1. Partially, because clouds and terrain are non-constant environment problems, clouds for obvious reasons and terrain because its influence depends on sensor and target locations. It is possible that handling these new environment effects

will require creating new objects below-the-line. At a minimum it will be necessary to modify the data fields associated with sensors.

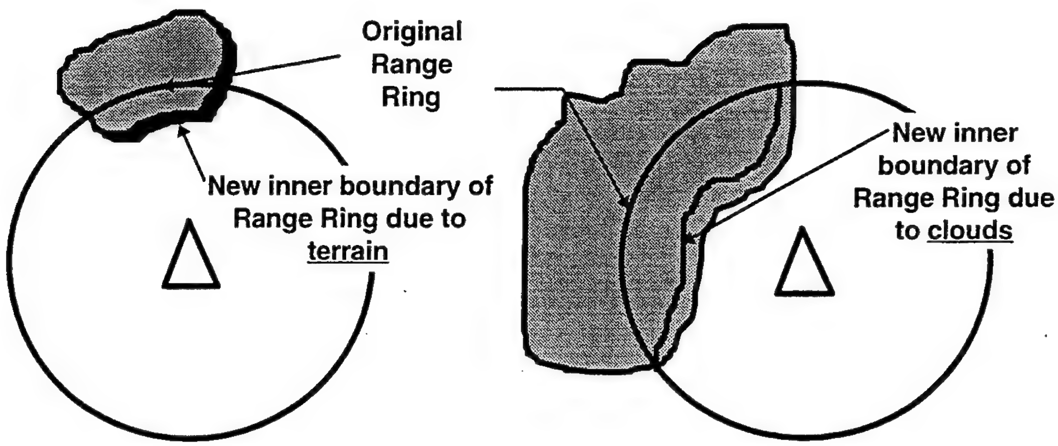


Figure 4.1. Altered Range Rings for Terrain and Clouds.

As mentioned, we want to implement the change in detection range caused by the earth's curvature. Both detector and target are not usually at sea level. Once these heights are taken into account, the range ring used by the simulation must be recalculated, now being a function of the detector's height (Z_d) and the target altitude (Z_t). See Figure 4.2. The current *NPS Platform Foundation* includes Z_t, but not Z_d. The detector data field contains only one entry, the detection range. It is proposed to substitute Z_d for this range, use a new method to calculate the horizon affected range, and then replace Z_d in the data field with this calculated range. This range is then available in the subsequent simulation processing. Following is the description of how this is accomplished.

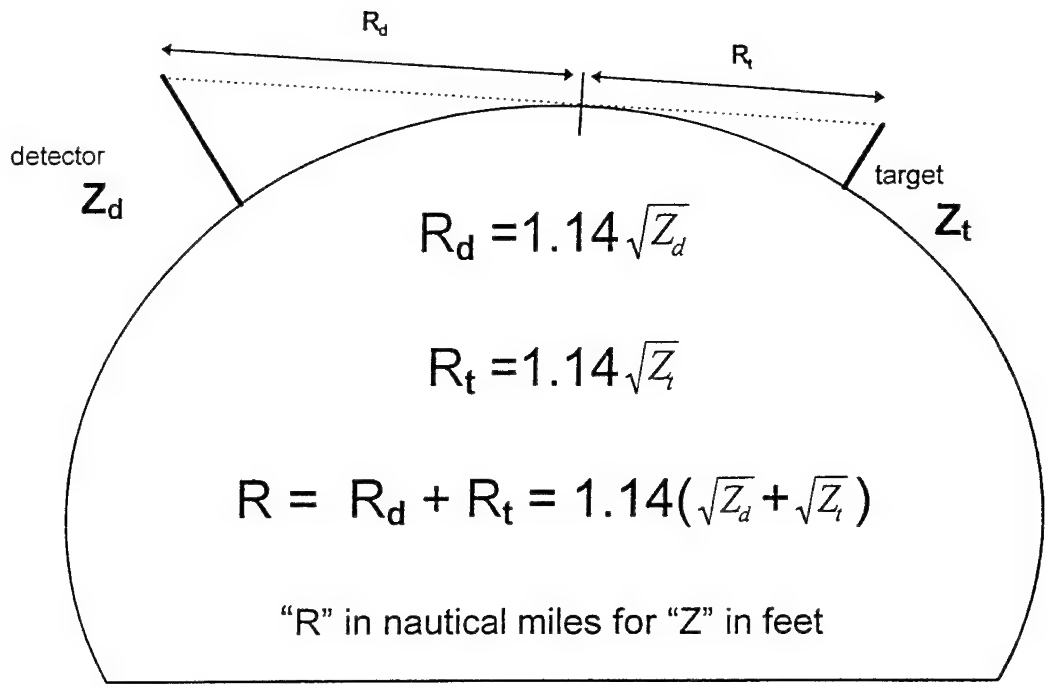


Figure 4.2. The Horizon Effect Calculation.

NPS Platform Foundation allows us to add or override generic methods to suit an application (in our case, change the range ring). The range formula will be implemented through the new Ask method, PrePlatformTEMPMETHOD. This new method will be inserted in the implementation module of TailoredSimExecutive (ITSimExc.mod). The method must be executed before the objects are instantiated because the range ring, R, must be available when the creation of the sensor occurs.

In Appendix B is shown the program source code used to implement the above methodology.

Notice that the above procedure cannot be extended to multi-target engagements, because each sensor can have available only one range distance regardless of the target. If a multi-threat comes into play, the solution for the horizon effect would need different detection ranges for each target height. Also, the target data file can assume only one constant value. If target altitudes are variable, this solution is not applicable because the range will vary during the course of the simulation. In order to include these variances, one would have to modify the sensor/platform data fields and the associated below-the-line attributes. There will be not a methodology proposed to accomplish these situations in this thesis.

V. CONCLUSION

The graphs presented in Chapter III show that optical systems can be modeled for target and detector characterization using the ANSI standard C programming language. The numerical outputs contained in the graphs were based upon the interaction of the system with its pertinent environment. The varying input parameters represented real-world situations for target-optical sensor systems. The physics of the radiant transfer problem was successfully segmented into the modules (functions) denoted in Appendix A. Each of these functions represents the solution for the respective physical aspect in the radiation problem.

Unfortunately, most military simulation tools are not designed to allow users to easily make modifications in order to incorporate real-life physical concepts. Very few of them offer details on sub-system interactions and performances. They were developed to focus on large scale engagements. To satisfy the requirement of this project proposal to simulate the performance of an optical detection system, it became necessary to work with software that allows objects configuration (tailoring) by adding or refining implementations of sensor responses. *NPS Platform Foundation* has proved to have the necessary versatile structure that enables the user to incorporate real-world situations into a simulation program.

For this thesis the horizon effect methodology was successfully implemented in *NPS Platform Foundation* through the below-the-line implementation module TailoredSimExecutive (ITSimExc.mod). Although accounting for the Earth's curvature is basic and simple, it demonstrates the feasibility of the approach. The result also indicates that extension of the methodology to many-on-many engagements and incorporation of more complicated environmental effects will require changes in the *NPS Platform Foundation* data structure and below-the-line program structure.

A primary goal of this thesis research, as stated in the Introduction, was to develop software modules to simulate the production and detection of infrared radiation and to incorporate those modules into a realistic simulation of military systems. The modules were successfully developed, so that goal was met. However, unexpected complications and a lack of time prevented their incorporation into *NPS Platform Foundation.* However, the successful incorporation of the Earth's curvature modification into *NPS Platform Foundation* has demonstrated that the *NPS Platform Foundation* simulation tool can be modified. It is left for follow-on thesis projects to focus on the best way to change the *NPS Platform Foundation* data structure, in order to account for the other environmental effects

APPENDIX A - THE "C" PROGRAM MODULES

In this Appendix we find the computer code, in ANSI standard C, for the basic modules that simulate radiation production, propagation and detection processes. There are two distinct sets of files. One set is for power detectors, and the other is for photon detectors. These files contain a number of functions (modules), with which each specific part of the characterization problem is resolved. Each module's listing contains comment lines that describe its function.

```
/*****************************************************************************/
/*                              PHOTON DETECTOR                              */
/*               This program returns data for photon detectors              */
/*            Source file name: photon.c                                     */
/*            Language:ANSI C                                                */
/*            Date: 20/APR/1995        Revised:25/JUL/1995                   */
/*****************************************************************************/

# include <stdio.h>
# include <math.h>
# define pi 3.141592654
float Simp(float,float,float);
float Mp(float,float);
float Emis(float);
float Noise(float,float,float,float);
float Flux(float,float,float,float,float,float,float);


float Simp(float llam, float ulam, float T)
{
/*************************************************************************/
/* llam - is the lower wavelength limit of integration, in meters        */
/* ulam - is the upper wavelength limit of integration, in meters        */
/* T - is the absolute temperature, in degrees Kelvin                    */
/* intervals - is the numbers of intervals(bins). Must be an even interger*/
/* The method of numerical integration is the Simpson's rule             */
/* The function "Mp" calculates the Photon flux exitance                 */
/*************************************************************************/
int n,intervals;
float S,h,x,y,C,D,E,a;
intervals=10000;
h=(ulam-llam)/intervals;
S=0.0;
a=llam;
C=Mp(a,T);
for(n=1;n<=intervals;n += 2)
{
     x=a+h;
     D=Mp(x,T);
     y=x+h;
     E=Mp(y,T);
     S += C + 4.0*D + E;
     a=y;
     C=E;
}
S *= h / 3.0 ;
return S;
}


/*************************************************************************/
/* This function(Mp), calculates the Photon flux exitance for blackbodies. */
/* It does NOT consider emissivity.                                      */
/*************************************************************************/
/*float Mp(float l,float T)
{
float Mp,two_pi_c,hck;
two_pi_c=1.88365e9;
hck=1.438761e-2;
```

```
Mp=l*l;
Mp=Mp*Mp;
Mp=two_pi_c/Mp;
return Mp/(exp(hck/(l*T))-1.0);
}*/


/**********************************************************/
/* "Emis", introduces the emissivity for non-blackbodies. */
/**********************************************************/
float Emis(float l)
{
return 1.0;
}


/**************************************************************************/
/* This function(Mp),calculates the Photon flux exitance for NON-blackbodies.*/
/*                      It considers emissivity.                          */
/**************************************************************************/
float Mp(float l,float T)
{
float Mp,two_pi_c,hck;
two_pi_c=1.88365e9;
hck=1.438761e-2;
Mp=l*l;
Mp=Mp*Mp;
Mp=two_pi_c/Mp;
return Emis(l)*(Mp/(exp(hck/(l*T))-1.0));
}


/**************************************************************************/
/*        This function(Noise),calculates the Photon flux exitance noise for */
/* NON-blackbodies(it considers emissivity). It is the noise(rms fluctuation)*/
/* of a target seen by a photon detector.                                  */
/**************************************************************************/
float Noise(float l1, float l2, float kelvin, float t)
{
float N;
N=Simp(l1,l2,kelvin);
return sqrt(N*t);
}


/**************************************************************************/
/*   This function(Flux), calculates the signal incident on the detector.     */
/* It considers the angle between the detector's plane normal and the         */
/*source-detector line is zero(teta.d=0).                                      */
/*   Sa = area of target(source) in meters square.                             */
/*   Da = area of detector in meters square.                                   */
/*   Steta = angle between target's plane normal and source-detector line in  */
/* radians.                                                                    */
/*   range = distance between source(target) and detector in meters.          */
/**************************************************************************/
float Flux (float la, float lb, float vinkel, float Sa, float Da, float Steta,
  float range)
{
float F;
F=(Simp(la,lb,vinkel)*Sa*Da*cos(Steta));
```

```
return F/(range*range*pi);
}
```

```
/***********************************************************************/
/*                         ENERGY DETECTOR                             */
/*            This program returns data for energy detector            */
/*          Source file name: energy.c                                 */
/*          Language:ANSI C                                            */
/*          Date: 19/APR/1995        Revised:09/AGO/1995               */
/***********************************************************************/

# include <stdio.h>
# include <math.h>
# define pi 3.141592654
# define q 1.60e-19
float Simp(float,float,float);
float Me(float,float);
float Flux(float,float,float,float,float,float,float);
float Emis(float);
float Irms(float,float,float,float,float,float,float,float,float);



float Simp(float llam, float ulam, float T)
{
int n,intervals;
float S,h,x,y,C,D,E,a;
/************************************************************************/
/* llam - is the lower wavelength limit of integration, in meters      */
/* ulam - is the upper wavelength limit of integration, in meters      */
/* T - is the absolute temperature, in degrees Kelvin                  */
/* intervals - is the numbers of intervals(bins). Must be an even interger*/
/* The method of numerical integration is the Simpson's rule           */
/* The function "Me" calculates the Planck distribution                */
/************************************************************************/
intervals=10000;
h=(ulam-llam)/intervals;
S=0.0;
a=llam;
C=Me(a,T);
for(n=1;n<=intervals;n += 2)
{
     x=a+h;
     D=Me(x,T);
     y=x+h;
     E=Me(y,T);
     S += C + 4.0*D + E;
     a=y;
     C=E;
}
S *= h / 3.0 ;
return S;
}

/************************************************************************/
/* This function(Me), calculates the Planck's distribution for blackbodies.*/
/* It does NOT consider emissivity.                                     */
/************************************************************************/
/*float Me(float l,float T)
{
float Me,pihcc,hck;
```

```
pihcc=3.741755e-16;
hck=1.438761e-2;
Me=l*l;
Me=Me*Me*l;
Me=pihcc/Me;
return Me/(exp(hck/(l*T))-1.0);
}*/


/*********************************************************/
/* "Emis", introduces the emissivity for non-blackbodies. */
/*********************************************************/
float Emis(float l)
{
return 1.0;
}


/*****************************************************************************/
/* This function(Me),calculates the Planck's distribution for NON-blackbodies*/
/*                        It considers emissivity.                           */
/*****************************************************************************/
float Me(float l,float T)
{
float Me,pihcc,hck;
pihcc=3.741755e-16;
hck=1.438761e-2;
Me=l*l;
Me=Me*Me*l;
Me=pihcc/Me;
return Emis(l)*(Me/(exp(hck/(l*T))-1.0));
}


/*****************************************************************************/
/*   This function(Flux), calculates the signal incident on the detector.    */
/* It considers the angle between the detector's plane normal and the        */
/*source-detector line is zero(teta.d=0).                                    */
/*   Sa = area of target(source) in meters square.                           */
/*   Da = area of detector in meters square.                                 */
/*   Steta = angle between target's plane normal and source-detector line in */
/* radians.                                                                  */
/*   range = distance between source(target) and detector in meters.        */
/*****************************************************************************/
float Flux (float la, float lb, float vinkel, float Sa, float Da, float Steta,
  float range)
{
float F;
F=(Simp(la,lb,vinkel)*Sa*Da*cos(Steta));
/*printf("F=%e\n",F);*/
return F/(range*range*pi);
}


/*****************************************************************************/
/*    The function "Irms", calculates the power flux noise(shot noise) for   */
/*NON-blackbodies(it considers emissivity). Shot noise is the dominant noise */
/*source for energy detectors.                                               */
/*          q is the elementary charge constant;                             */
/*          Ri is the detectors' responsivity;                               */
```

```
/*         flux is the power flux;                                              */
/*         tempo is the carrier transit time;                                   */
/*******************************************************************************/
float Irms (float li,float ls,float deg_kel,float aa,float ar_det,float angle,
float dist,float Ri,float tempo)
{
float n;
n=Flux(li,ls,deg_kel,aa,ar_det,angle,dist);
return sqrt(q*Ri*n/tempo);
}
```

APPENDIX B - THE *NPS PLATFORM FOUNDATION* IMPLEMENTATIONS

This Appendix describes the modifications made in the implementation module TailoredSimExecutive (ITSimExc.mod). Below is the actual code used to implement the horizon effect.

```
{-----------------------------------------------------------------------------------------------------------
}
ASK METHOD PreplatformTEMPMETHOD;
{-----------------------------------------------------------------------------------------------------------

}

VAR

        R,

        SensorHeight,

        TargetHeight : REAL;

        mySensorRec : SensorInfoRecType;

        myPlatformRec : PlatformInfoRecType;

BEGIN

        mySensorRec := ASK MasterSensorList First ( ) ;

        OUTPUT;

        OUTPUT;

        OUTPUT ( "Sensor DefaultRange on Input : ",

mySensorRec.DefaultRange ) ;

        myPlatformRec := ASK MasterPlatformInfoList First ( ) ;
```

```
        OUTPUT;
        OUTPUT ( "PlatformHeight = ", myPlatformRec.InitialLocation.z ) ;
        SensorHeight := mySensorRec.DefaultRange ;
        TargetHeight := myPlatformRec.InitialLocation.z ;
        R := 1.14 * ( SQRT ( SensorHeight ) + SQRT ( TargetHeight ) ) ;
        mySensorRec.DefaultRange := R;
        OUTPUT ( "Sensor Default Range is now : ", mySensorRec.DefaultRange
) ;
END METHOD ; .
```

In order to compile the modified ITSimExc.mod, the following must be imported:

- From MathMod, SQRT.
- From DetRng, SensorInfoRecType and MasterSensorList.
- From PList, PlatformInfoRecType and MasterPlatformInfoList.
- From GM, LocationRecType.

LIST OF REFERENCES

Bailey, M. P., *The NPS Platform Foundation*, Technical Paper, Department of Operations Research, Naval Postgraduate School, Monterey, California, February 1994.

Belanger, R., *MODSIM II - A High-Level, Object-Oriented Language*, Technical Paper, ErgoSoft, 7122 Cather Ct., San Diego, California, 1993.

Born, M. and Wolf, E., *Principles of Optics*, Pergamon Press, Sixth Edition, 1980.

Boyd, R. W., *Radiometry and the Detection of Optical Radiation,* J. Wiley & Sons, 1983.

Chandrasekhar, S., *Radiative Transfer*, Dover, 1960.

Dereniak, E. L. and Crowe, D. G., *Optical Radiation Detectors*, J. Wiley & Sons, 1984.

Fishwick, P. A., *Computer Simulation: Growth Through Extension*, Technical Paper, Computer and Information Sciences Department, University of Florida, CSE 301, April 1994.

Goody, R. M. and Yung, Y. L., *Atmospheric Radiation*, Oxford University Press, Second Edition, 1989.

Hecht, E., *Optics*, Addison-Wesley Publishing Company, Second Edition, February 1989.

Wolfe, W. L. and Zissis, G. J., *The Infrared Handbook*, Revised Edition, ERIM, 1989.

INITIAL DISTRIBUTION LIST

1. Defense Technical Information Center 2
   8725 John J. Kingman Rd., STE 0944
   Ft. Belvoir, VA 22060-6218

2. Library, Code 13 2
   Naval Postgraduate School
   Monterey, CA 93943-5101

3. Chairman, Department of Physics.. 1
   Code PH
   Naval Postgraduate School
   Monterey, CA 93943-5100

4. Prof. David Scott Davis 1
   Code PH/Dv
   Naval Postgraduate School
   Monterey, CA 93943

5. Prof. Gordon Everett Schacher 1
   Code PH/Sq
   Naval Postgraduate School
   Monterey, CA 93943

6. LCDR Carlos Renato Campos Rangel.. 2
   1061 Spruance Rd
   Monterey, CA 93940